Introdução Git

quinta-feira, 14 de abril de 2022

04:06

Git é um software de versionamento de código. Ele ajuda a criar e monitorar diferentes versões de um código.

Ele é um CLI (command-line intarfece), diferente do GUI (graphical user interface)

Comandos de terminal

Windows

* cd - Navega entre as pastas
* dir - Listar diretórios da pasta.
* mkdir - Criar um diretório
* echo conteudo > arquivo.extensao - Criar um arquivo
* del/rmdir - deletar arquivo/diretório
* cls - limpar tela

Unix (variações de Linux e Apple)

* cd
* ls
* Mkdir
* echo conteudo > arquivo.extensao
* rm -rf
* clear

Instalação

Windows:

1. <https://git-scm.com/download/win>
2. 
3. Instalar

Funcionamento Git

quinta-feira, 14 de abril de 2022

04:56

Tópicos

* SHA1
* Objetos fundamentais

SHA1

SHA - Secure Hash Algorithm (Algoritmo de Hash Seguro), é um conjunto de funções hash criptográficas projetadas pela NSA (Agência de Segurança Nacional dos EUA).

Ele basicamente é um algoritmo de encriptação. A grosso modo, ele pega um arquivo e embaralha de uma forma muito específica.

A encriptação gera um conjunto de caracteres identificador de 40 dígitos. Ou seja, um arquivo que tenha 800 linhas de código, ele gerará um código de 40 dígitos. Se algo for alterado, será gerado um código diferente.

**Isso serve para que o Git identifique se um arquivo foi modificado ou não.**

Como usar

1 - Tenha um arquivo qualquer. Teste.txt por exemplo.

2 - Iniciar o Git Bash na pasta do arquivo.

3 - openssl sha1 teste.txt

Objetos Fundamentais

**Blobs**

Cada arquivo no Git é armazenado como um objeto blob. Ou seja, esses arquivos possuem metadados próprios, diferente de arquivos comuns fora do git.

Exemplo:

O Git possuí uma função própria para gerar um sha1, através do comando "git hash-object". Mas também é possível utilizar o algoritmo anteriormente visto "openssl sha1".

A diferença será, que: ao utilizar esses comandos em arquivos ou strings iguais, o SHA1 gerado será diferente, justamente pelo openssl sha1 não identificar esses metadados do Blobs. Já a função própria do git, sabe identificá-los, gerando outro SHA1.

Para que o openssl sha1 gere o mesmo SHA1 da função do git, é preciso especificar os metadados, já que eles não são identificados como na função.

1 
2 
3 
4 
5 
echo 'conteudo' I git hash-object 
--stdin 
> fc31e91b26cf85a55e072476de7f263c8926eeb1 
echo -e 'blob 9\0conteudo' I openssl shal 
> fc31e91b26cf85a55e072476de7f263c8926eeb1 

**Trees**

As trees armazenam as Blobs. Essas árvores também possuem metadados, além de apontar para blobs, que por sua vez possuí um SHA1 e o nome do arquivo.

É como se fossem diretórios, onde cada pasta é uma árvore, que possuem blobs que seriam os arquivos. Sendo assim, Trees também apontam pra outras Trees e Blobs.

blob 
tree 
blob 
tree 
Imp legit.rö 
b tob 

**Commits**

É o objeto que junta tudo. O commit aponta para uma arvore, para o parente (ultimo commit feito) e para o autor, além de uma mensagem e o horário de criação.

SHAI 487d4s 
ommlt 
ree 
arente 
utor 
mensagem 
imestamp 
s4a5sq1 
a98acq1 
perkles 
"inicia ..." 

Ao final, o esquema é o seguinte:

Se um blob for alterado, seu SHA1 será alterado. Esse blob está sendo apontado por uma arvore, ou seja, os metadados dessa arvore serão alterados, alterando também seu SHA1. Por fim, essa árvore é apontada por um Commit. É uma forma EXTREMAMENTE segura para identificar alterações.

98ca9 
size 
tree 
size 
tree 92ec2 
btOb 5b1d3 
author Scott 
blOb 911e7 
comitter Scott 
cba0a 
•nit lat O' project 
REA(YE 
LICENSE 
test. rb 
Testing tibrary 
This library to test 
Ruby prolects. 
91 re? 
siz- 
The License 
Copyright Scott Chaco' 
requ re 
haøa 
• test 'unit 

Chaves SSH e Tokens

quinta-feira, 14 de abril de 2022

05:40

Chave SSH

Chave SSH é uma forma de estabelecer uma conexão segura e encriptada entre duas máquinas. No caso, seu computador com o servidor do Github.

Haverá duas chaves, a pública e a privada. A pública é enviada ao Github, confirmando que a conexão é segura e que se trata de fato da sua máquina.

Para criar uma chave SSH:

**Criando**

1. Abrir o Git Bash
2. ssh-keygen -t ed25519 -C email@email.com
3. Dê enter e insira uma senha
4. cd /c/Users/User/.ssh/
5. cat nome da chave
6. Copie a chave exibida

**Enviando**

1. Acessar o Github
2. Ir em configurações -> SSH and GPG keys
3. New SSH key
4. Insira um nome e cole a chave

**Ativando**

1. Abrir Git Bash
2. eval $(ssh-agent -s)
3. ssh-add nome da chave **PRIVADA (sem o .pub)**
4. Insira a senha da chave

Token de Acesso Pessoal

Token é uma chave de uso pessoal, que deverá ser informada quando for utilizar um repositório. Para saber mais, acessar:

<https://docs.github.com/pt/enterprise-cloud@latest/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token>

Comandos Git

quinta-feira, 14 de abril de 2022

11:37

Iniciando

* Abra o git bash na pasta de um projeto
* git init (inicia um repositório)
* git config --global user.email "email@email.com"
* git config --global user.name nome

Após isso, crie seus arquivos.

Commit

* git add \*
* git commit -m "alguma mensagem"

Esses comandos são responsáveis por alterar o estado dos arquivos no git

Untracked 
Unmodified 
Tracked 
Modified 
•Stage" o arqLivo 
Commit 
Staged 
Adiciona o arquivo 
Edita 0 a rquivo 
Remove 0 arquivo 

Untracked são os arquivos que o git ainda não conhece. (O arquivo deve ser commitado para que o git reconheça)

Tracked são os arquivos que o git já conhece.

* Unmodified são arquivos que já foram commitados e não foram mais alterados.
* Modified são os arquivos que sofreram alterações.
* Staged são os arquivos que foram citados no comando git add, ou seja, adicionados ao staged, apenas esperando para serem commitados.

Cada commit vai para um repositório local, que posteriormente pode ser enviado para um repositório remoto (github)

Lembrando que commit é um objeto que aponta para trees e blobs que sofreram alterações, além do commit anterior a ele.

Github

quinta-feira, 14 de abril de 2022

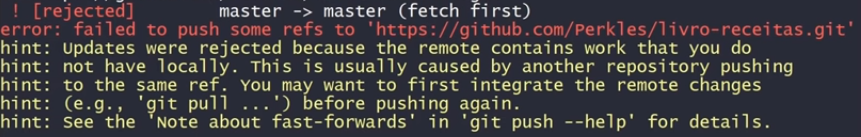
13:06

Adicionar arquivos a um repositório no Github

**git remote add origin link-do-repositorio** - define que o link se chama origin

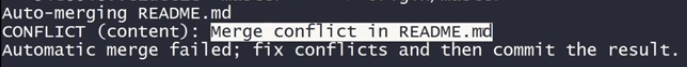
**git push origin main** - envia os arquivos

Possíveis Problemas



Provavelmente algum arquivo do repositório local está diferente ou faltando. Para isso use

**git pull origin main** - Puxar os arquivos mais recentes do repo. Remoto.



Isso se dá quando ele não consegue juntar dois arquivos e pede pra você resolver. Basta ir no arquivo e alterar do modo que deseja, e depois dar um add, commit e push.

Clonar um repositório

**git clone link-repositorio**